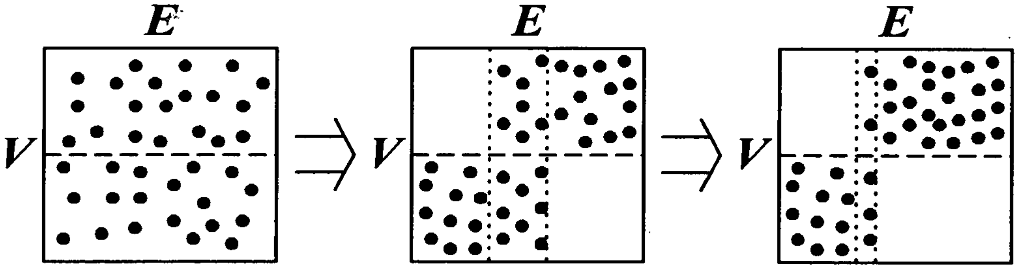


Fig. 1

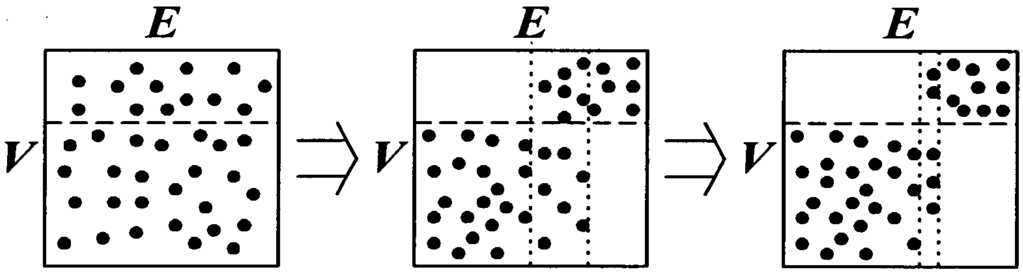


Fig. 2

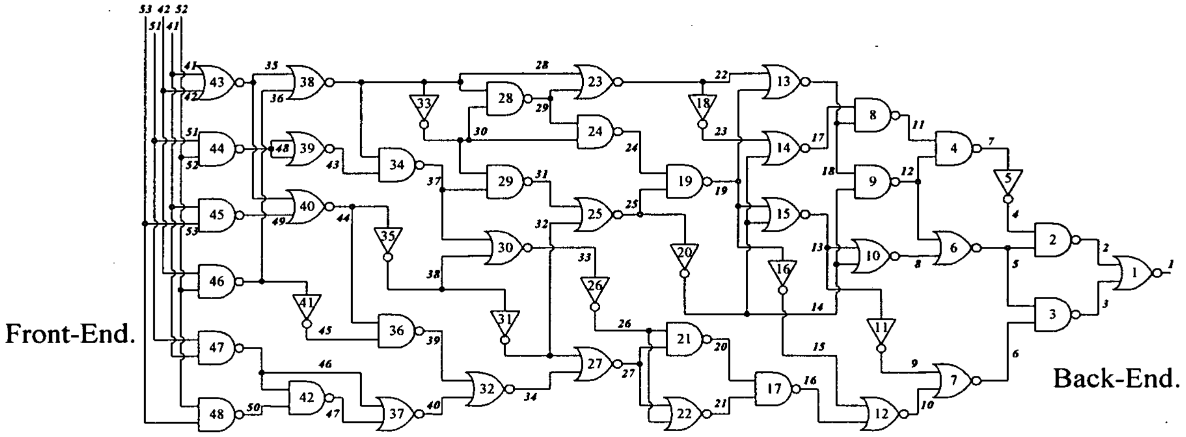

53 42 52
51 41
Front-End.
Back-End.
43
44
45
46
47
48
38
39
40
41
42
34
35
36
37
28
33
29
30
31
32
23
24
25
26
27
19
18
20
21
22
17
13
14
15
16
12
8
9
10
11
4
5
6
7
2
3
1

*Fig. 3*

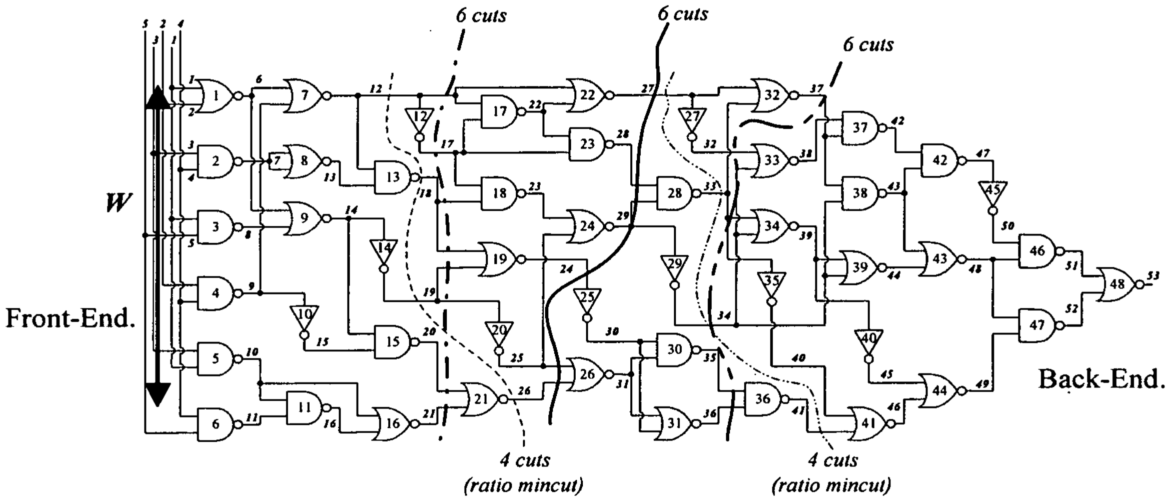

6 cuts
6 cuts
6 cuts
W
Front-End.
Back-End.
4 cuts
(ratio mincut)
4 cuts
(ratio mincut)

*Fig. 4*

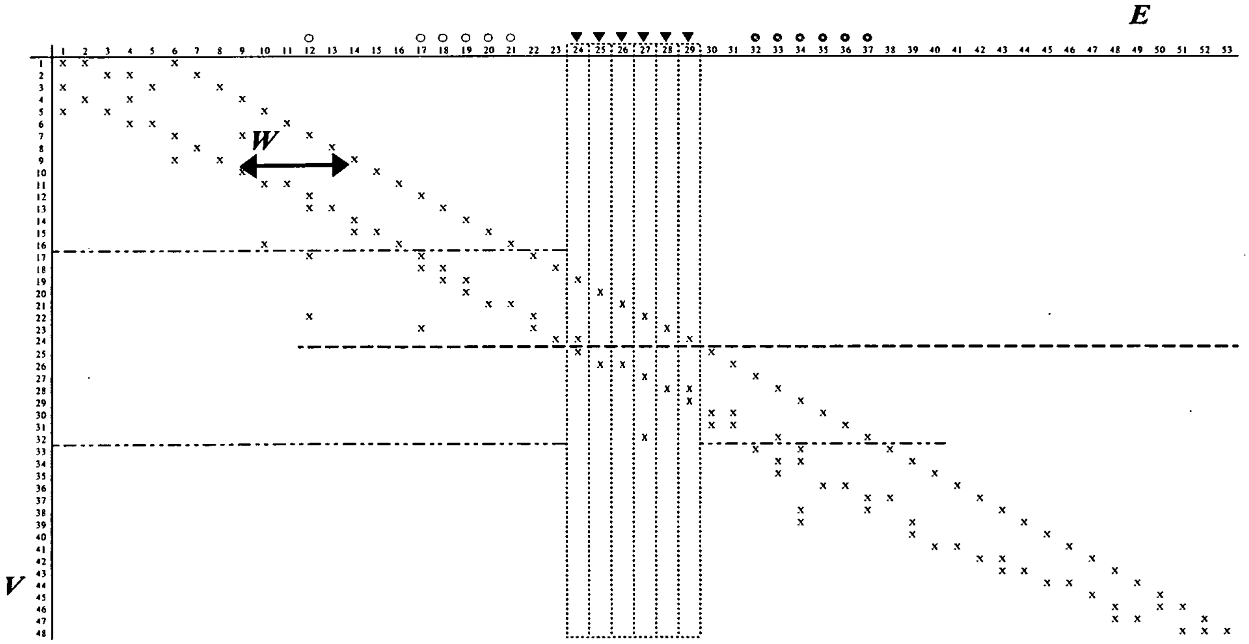

E
W
V

*Fig. 5*

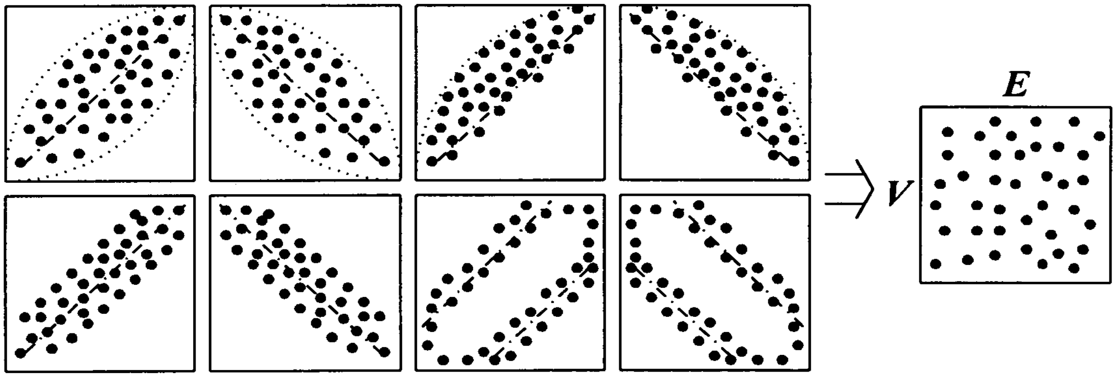
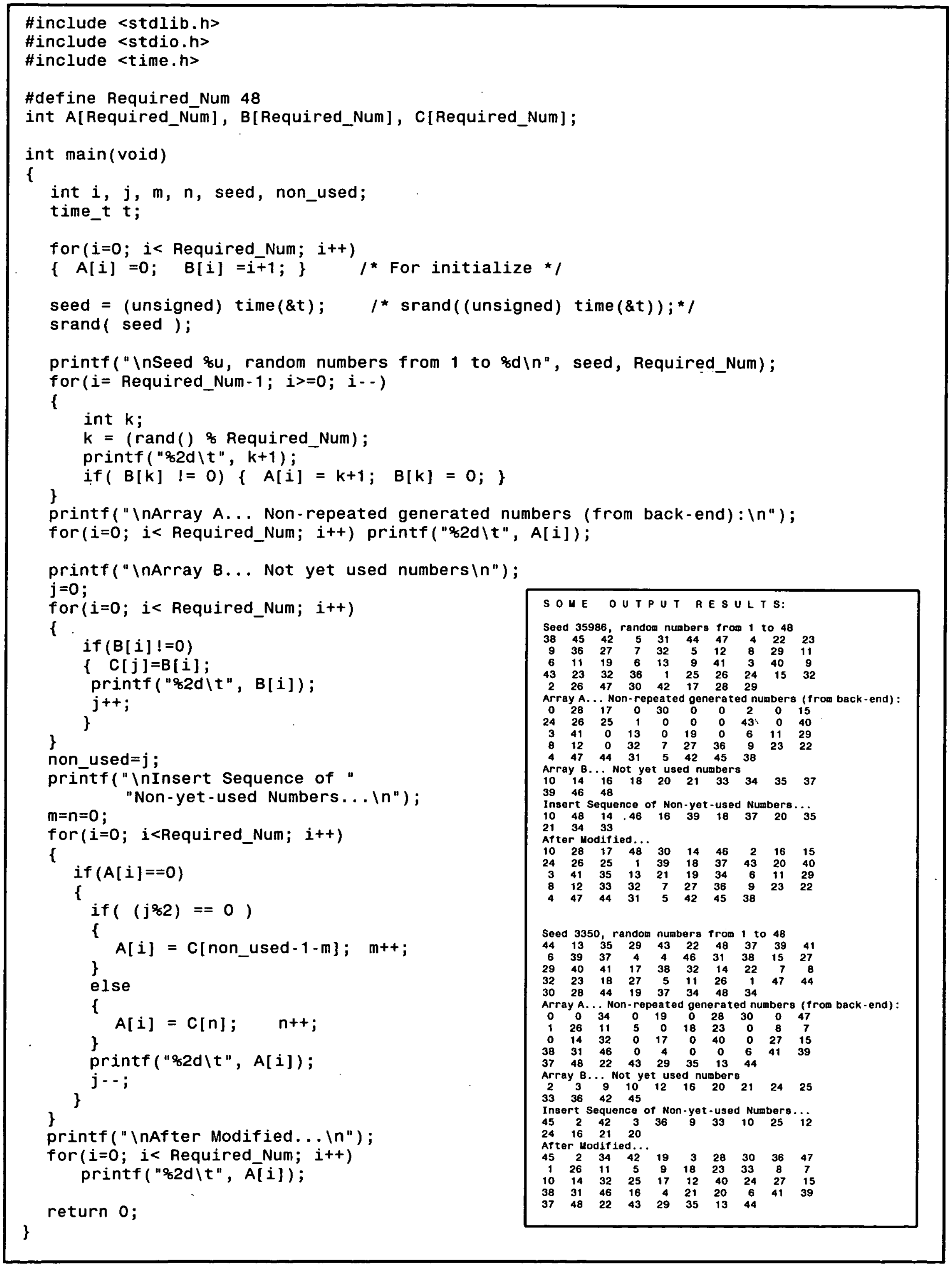


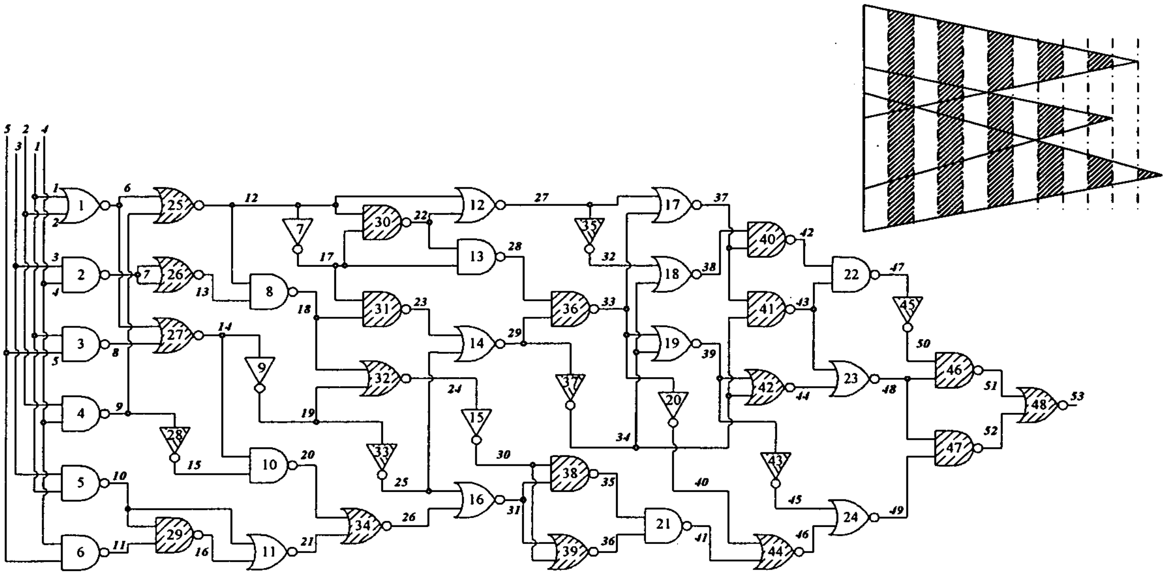
*Fig. 6*



```
#include <stdlib.h>
#include <stdio.h>
#include <time.h>

#define Required_Num 48
int A[Required_Num], B[Required_Num], C[Required_Num];

int main(void)
{
   int i, j, m, n, seed, non_used;
   time_t t;

   for(i=0; i< Required_Num; i++)
   {  A[i] =0;   B[i] =i+1; }      /* For initialize */

   seed = (unsigned) time(&t);     /* srand((unsigned) time(&t));*/
   srand( seed );

   printf("\nSeed %u, random numbers from 1 to %d\n", seed, Required_Num);
   for(i= Required_Num-1; i>=0; i--)
   {
       int k;
       k = (rand() % Required_Num);
       printf("%2d\t", k+1);
       if( B[k] != 0) {  A[i] = k+1;  B[k] = 0; }
   }
   printf("\nArray A... Non-repeated generated numbers (from back-end):\n");
   for(i=0; i< Required_Num; i++) printf("%2d\t", A[i]);

   printf("\nArray B... Not yet used numbers\n");
   j=0;
   for(i=0; i< Required_Num; i++)
   {
       if(B[i]!=0)
       {  C[j]=B[i];
        printf("%2d\t", B[i]);
        j++;
       }
   }
   non_used=j;
   printf("\nInsert Sequence of "
            "Non-yet-used Numbers...\n");
   m=n=0;
   for(i=0; i<Required_Num; i++)
   {
      if(A[i]==0)
      {
        if( (j%2) == 0 )
        {
           A[i] = C[non_used-1-m];  m++;
        }
        else
        {
           A[i] = C[n];    n++;
        }
        printf("%2d\t", A[i]);
        j--;
      }
   }
   printf("\nAfter Modified...\n");
   for(i=0; i< Required_Num; i++)
       printf("%2d\t", A[i]);

   return 0;
}
```

```
SOME OUTPUT RESULTS:

Seed 35986, random numbers from 1 to 48
38   45   42    5   31   44   47    4   22   23
 9   36   27    7   32    5   12    8   29   11
 6   11   19    6   13    9   41    3   40    9
43   23   32   36    1   25   26   24   15   32
 2   26   47   30   42   17   28   29
Array A... Non-repeated generated numbers (from back-end):
 0   28   17    0   30    0    0    2    0   15
24   26   25    1    0    0    0   43    0   40
 3   41    0   13    0   19    0    6   11   29
 8   12    0   32    7   27   36    9   23   22
 4   47   44   31    5   42   45   38
Array B... Not yet used numbers
10   14   16   18   20   21   33   34   35   37
39   46   48
Insert Sequence of Non-yet-used Numbers...
10   48   14  .46   16   39   18   37   20   35
21   34   33
After Modified...
10   28   17   48   30   14   46    2   16   15
24   26   25    1   39   18   37   43   20   40
 3   41   35   13   21   19   34    6   11   29
 8   12   33   32    7   27   36    9   23   22
 4   47   44   31    5   42   45   38


Seed 3350, random numbers from 1 to 48
44   13   35   29   43   22   48   37   39   41
 6   39   37    4    4   46   31   38   15   27
29   40   41   17   38   32   14   22    7    8
32   23   18   27    5   11   26    1   47   44
30   28   44   19   37   34   48   34
Array A... Non-repeated generated numbers (from back-end):
 0    0   34    0   19    0   28   30    0   47
 1   26   11    5    0   18   23    0    8    7
 0   14   32    0   17    0   40    0   27   15
38   31   46    0    4    0    0    6   41   39
37   48   22   43   29   35   13   44
Array B... Not yet used numbers
 2    3    9   10   12   16   20   21   24   25
33   36   42   45
Insert Sequence of Non-yet-used Numbers...
45    2   42    3   36    9   33   10   25   12
24   16   21   20
After Modified...
45    2   34   42   19    3   28   30   36   47
 1   26   11    5    9   18   23   33    8    7
10   14   32   25   17   12   40   24   27   15
38   31   46   16    4   21   20    6   41   39
37   48   22   43   29   35   13   44
```

*Fig. 7*



# Fig. 8A

```
Seed 34731, random numbers from 1 to 24
 1      10     21     8      17     6      4      7      22     15
 9      9      12     13     12     19     6      4      10     21
 23     11     4      24

Array A... Non-repeated generated numbers (from back-end)
 24     0      11     23     0      0      0      0      19     0
 13     12     0      9      15     22     7      4      6      17
 8      21     10     1

Array B... Not yet used numbers
 2      3      5      14     16     18     20

Insert Sequence of Non-yet-used Numbers...
 2      20     3      18     5      16     14

After Modified...
 24     2      11     23     20     3      18     5      19     16
 13     12     14     9      15     22     7      4      6      17
 8      21     10     1
```

```
Seed 34797, random numbers from 25 to 48
 33     41     28     40     33     45     36     48     44     39
 27     47     35     37     30     31     44     33     46     25
 35     28     30     46

Array A... Non-repeated generated numbers (from back-end)
 0      0      0      0      25     46     0      0      31     30
 37     35     47     27     39     44     48     36     45     0
 40     28     41     33

Array B... Not yet used numbers
 26     29     32     34     38     42     43

Insert Sequence of Non-yet-used Numbers...
 26     43     29     42     32     38     34

After Modified...
 26     43     29     42     25     46     32     38     31     30
 37     35     47     27     39     44     48     36     45     34
 40     28     41     33
```

# Fig. 8B

0. **Initialize:** mapping (V, E) pairs to V-E plain,
confirm the (V, E) pair distributed condition under nearly Max-cut reservation
and may randomize the node number order.

1. Phase One: basic four steps.

E N E N
(B) (R) (T) (L)

E: Edge Radix Sort
N: Node Radix Sort

( B ): Bottom-side base
( R ): Right-side base
( T ): Top-side base
( L ): Left-side base

2. Phase Two Begins: different additional steps can be choiced.

2A.
N E N
(R) (T) (L)

N E N
(R) (T) (L)

● ● ●

2B.
N E N N E N
(R) (T) (L) (R) (B) (L)

N E N N E N
(R) (T) (L) (R) (B) (L)

● ● ●

2C.
N E N E E N E N N E N E E N E N
(R) (T) (L) (B) (T) (L) (B) (R) (L) (B) (R) (T) (B) (R) (T) (L)

N E N E E N E N N E N E E N E N
(R) (T) (L) (B) (T) (L) (B) (R) (L) (B) (R) (T) (B) (R) (T) (L)

● ● ●

2D.
E N E N
(B) (R) (T) (L)

E N E N
(B) (R) (T) (L)

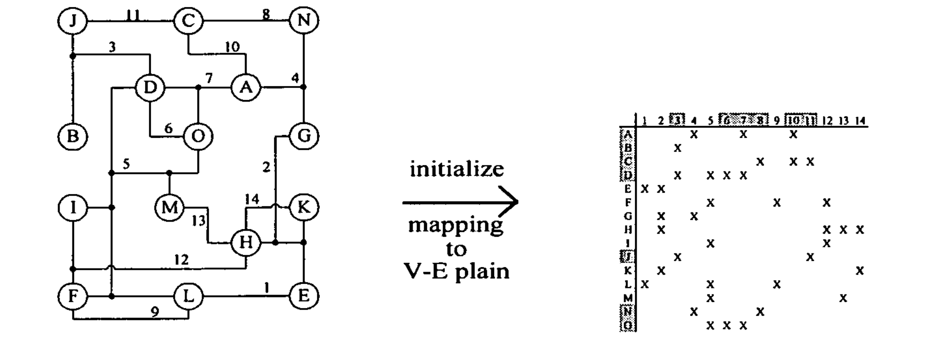
● ● ●

2E. Some other recurring orders.

2F. Some other clustering techniques.

**※When every sort step completed, record nodes set, and if node set no more change, halt the procedures.**

*Fig. 9*



A 14 edges / 15 nodes example.

Confirm the distributed condition.

*Fig. 10A*

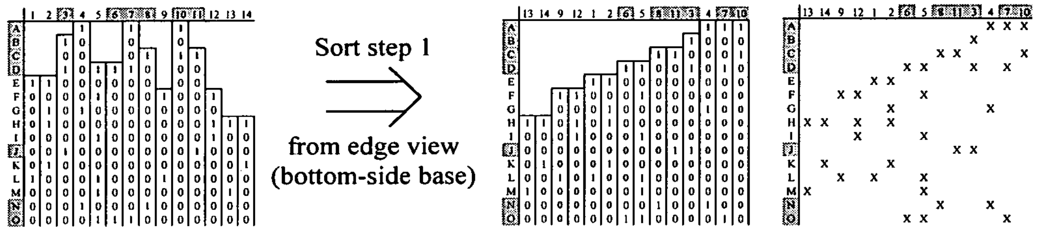


*Fig. 10B*

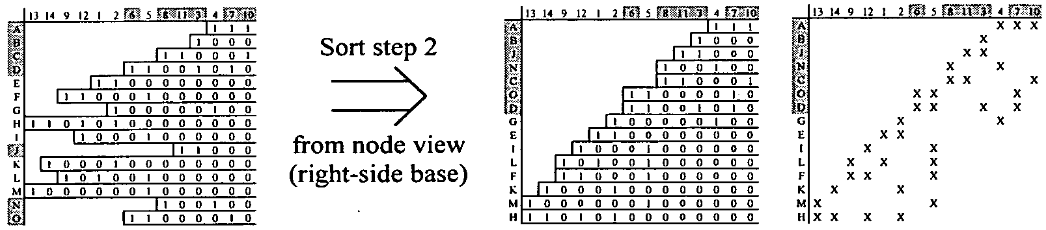
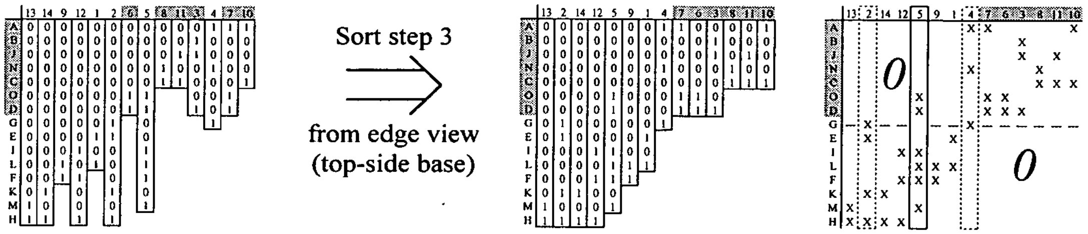
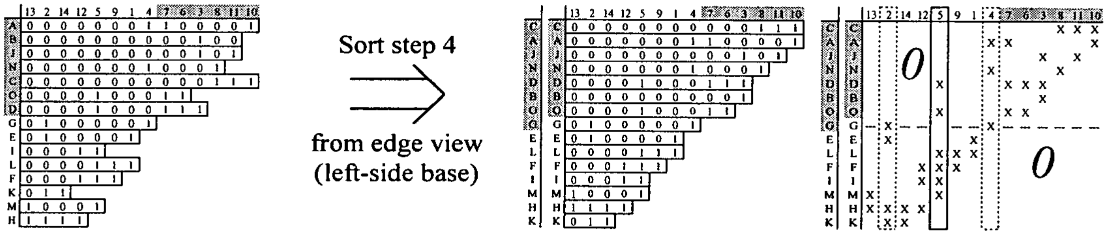


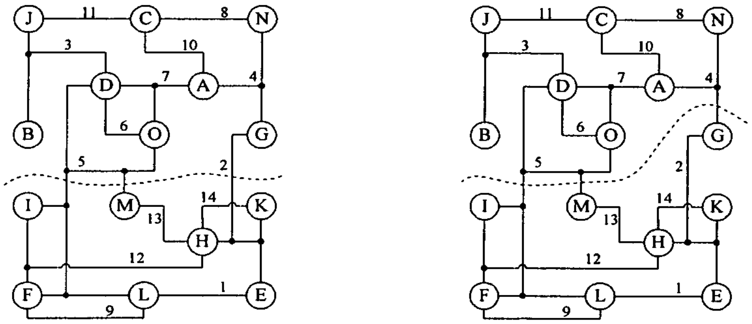
*Fig. 10C*



# Fig. 10D



# Fig. 10E



# Fig. 10F

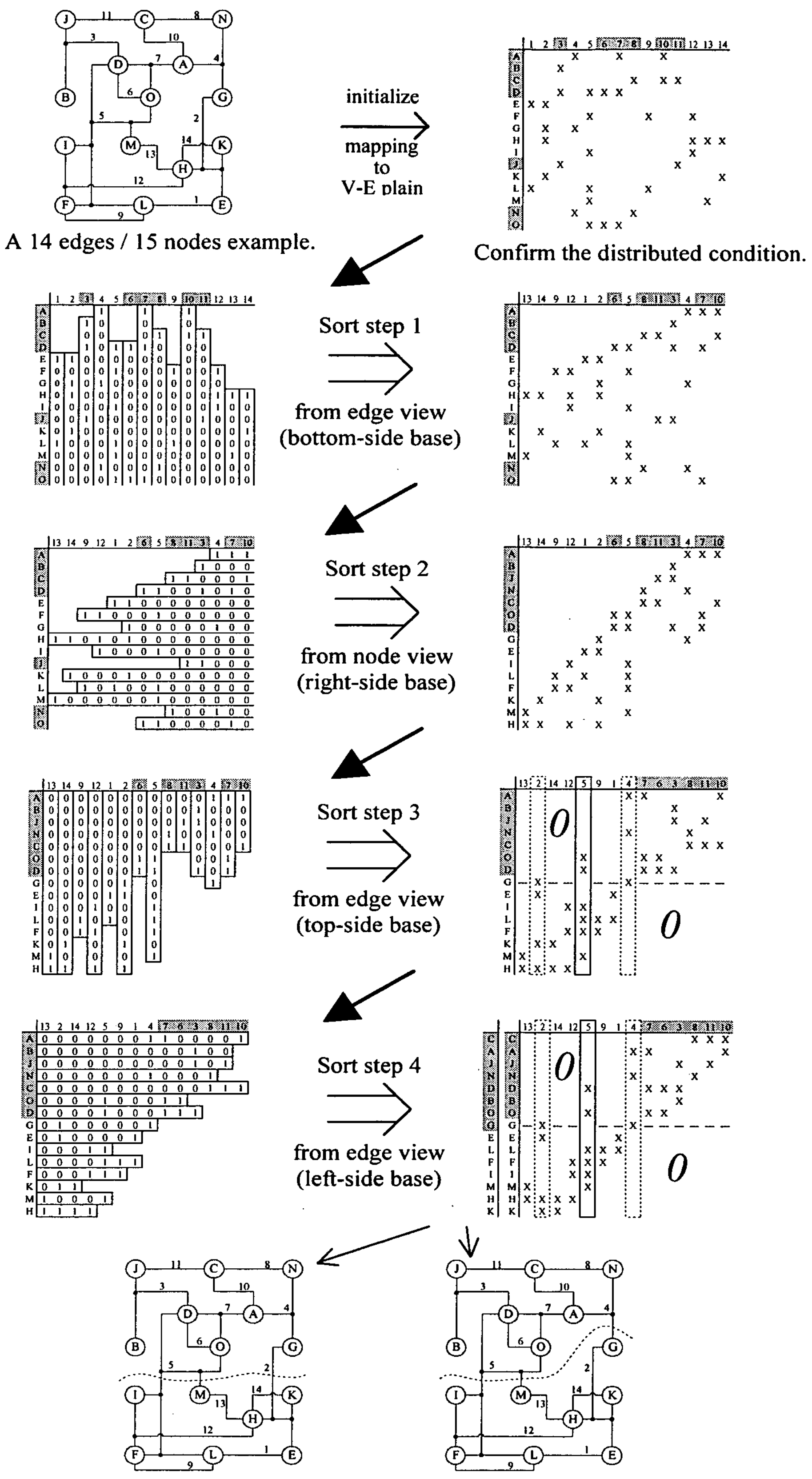
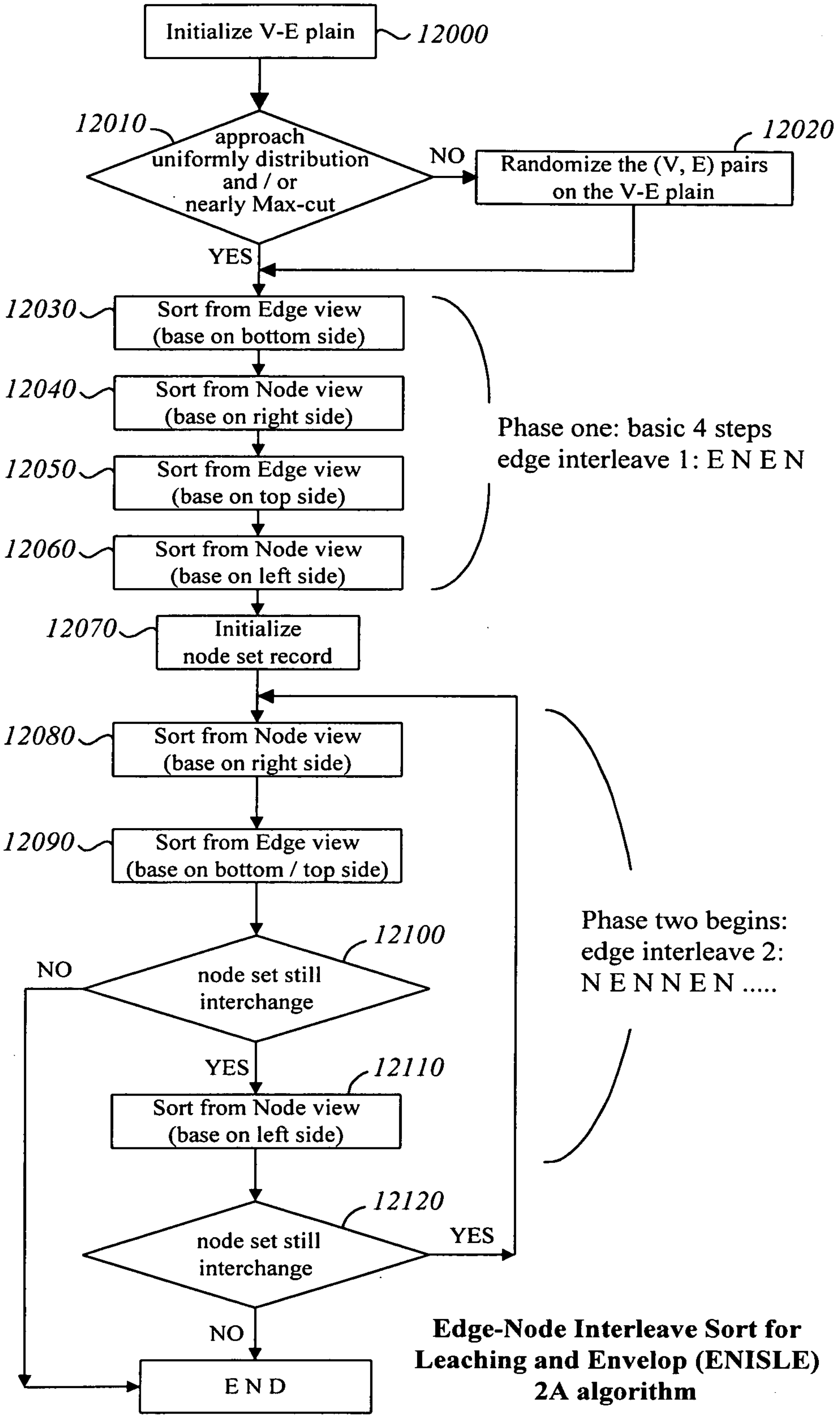
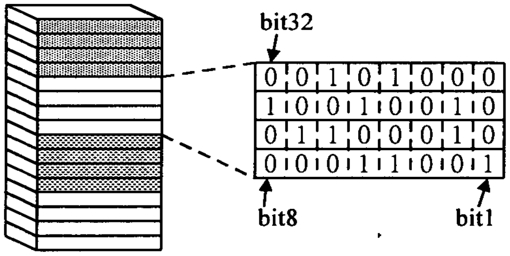

A 14 edges / 15 nodes example.
initialize
mapping
to
V-E plain
Confirm the distributed condition.
Sort step 1
from edge view
(bottom-side base)
Sort step 2
from node view
(right-side base)
Sort step 3
from edge view
(top-side base)
Sort step 4
from edge view
(left-side base)
0
0

*Fig. 11*


Initialize V-E plain
12000
12010
approach uniformly distribution and / or nearly Max-cut
NO
12020
Randomize the (V, E) pairs on the V-E plain
YES
12030
Sort from Edge view (base on bottom side)
12040
Sort from Node view (base on right side)
12050
Sort from Edge view (base on top side)
12060
Sort from Node view (base on left side)
Phase one: basic 4 steps edge interleave 1: E N E N
12070
Initialize node set record
12080
Sort from Node view (base on right side)
12090
Sort from Edge view (base on bottom / top side)
12100
NO
node set still interchange
YES
12110
Sort from Node view (base on left side)
Phase two begins: edge interleave 2: N E N N E N .....
12120
node set still interchange
YES
NO
E N D

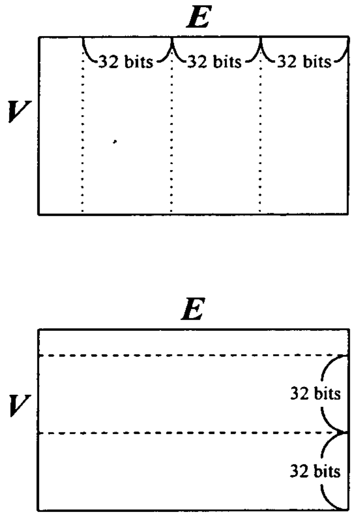
**Edge-Node Interleave Sort for Leaching and Envelop (ENISLE) 2A algorithm**

*Fig. 12*



```
struct bitfield32 {
     bit32 :1;
     bit31 :1;
     bit30 :1;
.......
     bit2 :1;
     bit1 :1;
} radix_sort_unit;
```

*Fig. 13*



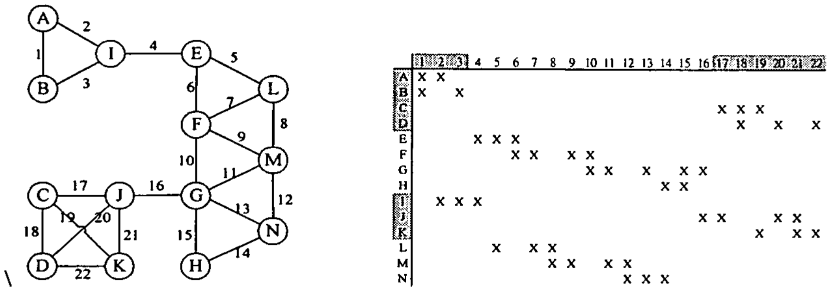
**Radix Sorting (LSD) Example:**

232, 321, 213, 231, 111, 112, 132, 123, 221
1S → 321, 231, 111, 221
2S → 232, 112, 132
3S → 213, 123

321, 231, 111, 221, 232, 112, 132, 213, 123
10S → 111, 112, 213
20S → 321, 221, 123
30S → 231, 232, 132

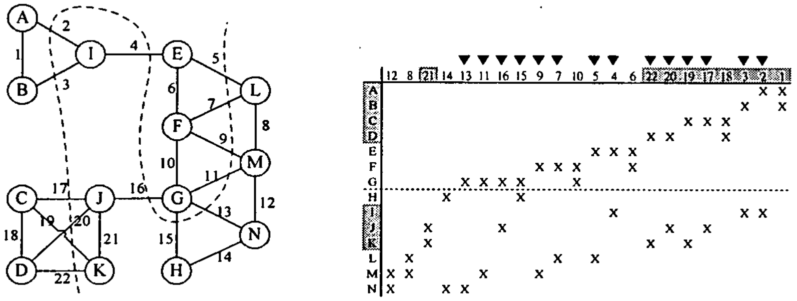
111, 112, 213, 321, 221, 123, 231, 232, 132
100S → 111, 112, 123, 132
200S → 213, 221, 231, 232
300S → 321
Output: 111, 112, 123, 132, 213, 221, 231, 232, 321

*Fig. 14*



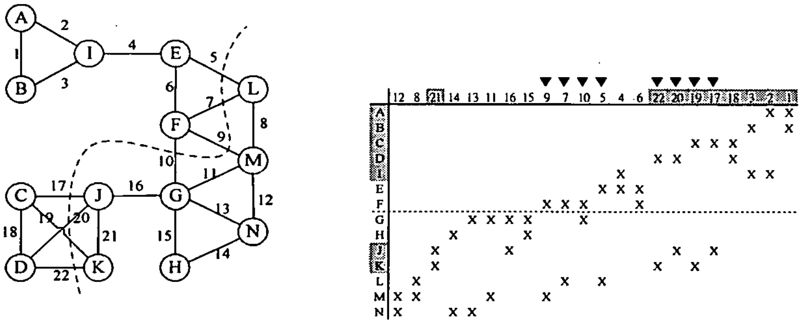
Initialize the V-E Plain.

*Fig. 15A*



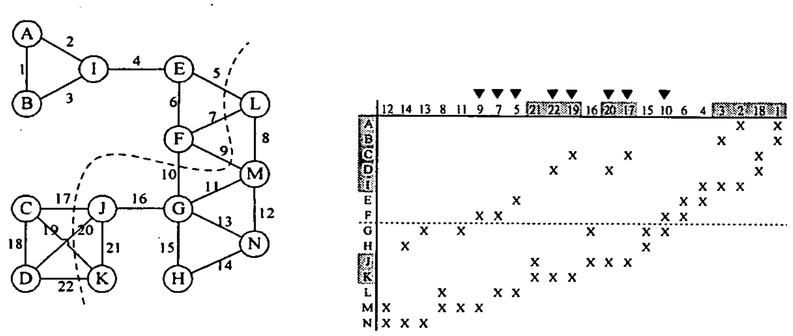
Step 1, cut numbers: 14.

*Fig. 15B*



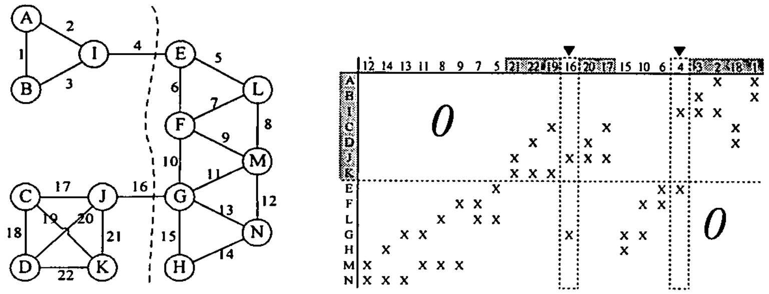
Step 2, cut numbers: 8.

*Fig. 15C*



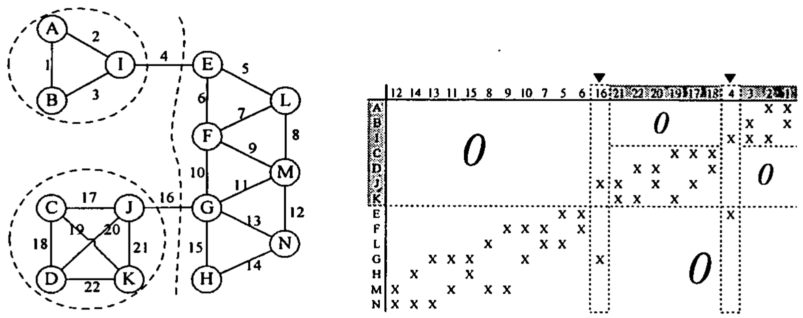
Step 3, 4, cut numbers: 8.

*Fig. 15D*



Step 5, cut numbers: 2.

*Fig. 15E*



Step 6, cut numbers: 2.

*Fig. 15F*

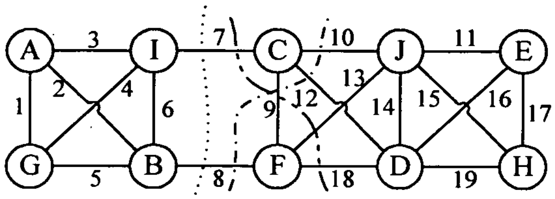


*Fig. 16*

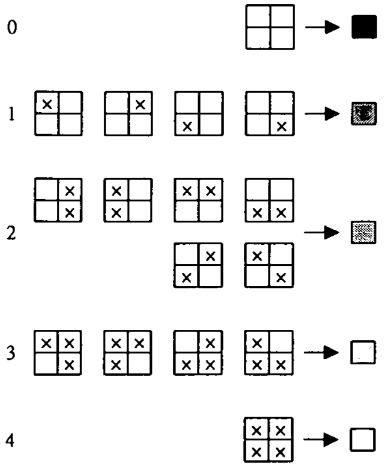


*Fig. 17A*

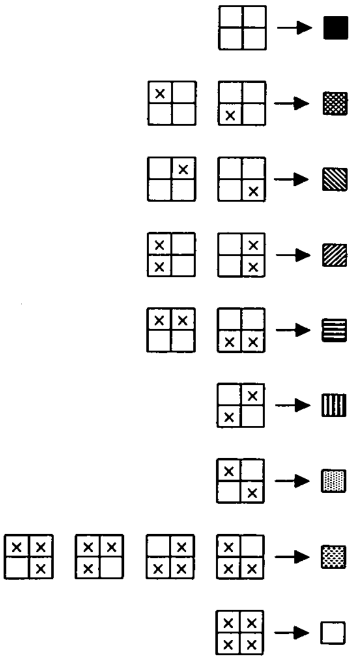


*Fig. 17B*

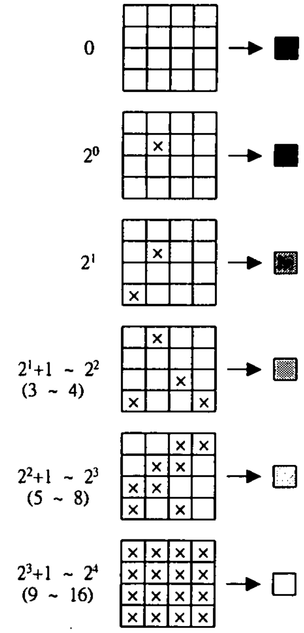


*Fig. 17C*

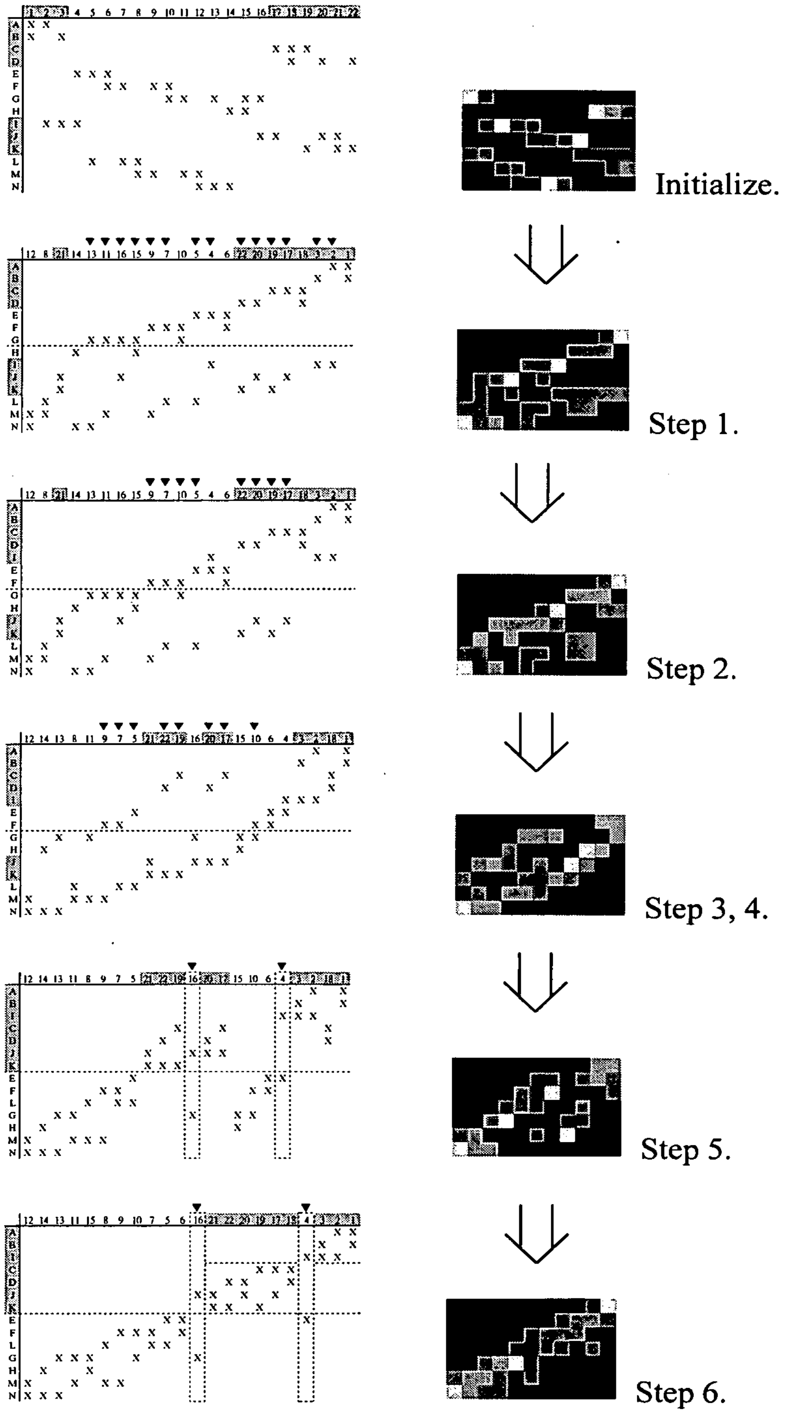
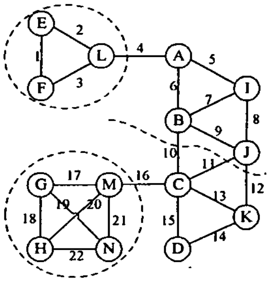


*Fig. 18.*



| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | | | | X | X | X | | | | | | | | | | | | | | | | |
| B | | | | | | X | X | | X | X | | | | | | | | | | | | |
| C | | | | | | | | | | X | X | | X | | X | X | | | | | | |
| D | | | | | | | | | | | | | | X | X | | | | | | | |
| E | X | X | | | | | | | | | | | | | | | | | | | | |
| F | X | | X | | | | | | | | | | | | | | | | | | | |
| G | | | | | | | | | | | | | | | | | X | X | X | | | |
| H | | | | | | | | | | | | | | | | | | X | | X | | X |
| I | | | | | X | | X | X | | | | | | | | | | | | | | |
| J | | | | | | | | X | X | | X | X | | | | | | | | | | |
| K | | | | | | | | | | | | X | X | X | | | | | | | | |
| L | | X | X | X | | | | | | | | | | | | | | | | | | |
| M | | | | | | | | | | | | | | | | X | X | | | X | X | |
| N | | | | | | | | | | | | | | | | | | | X | | X | X |

*Fig. 19*

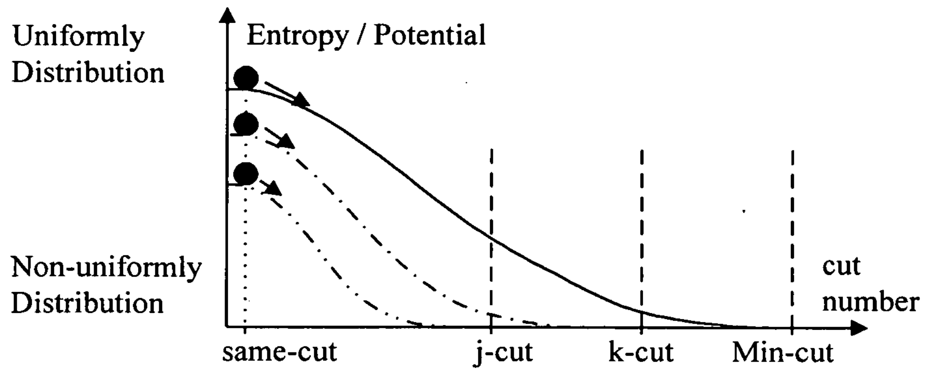


*Fig. 20A*

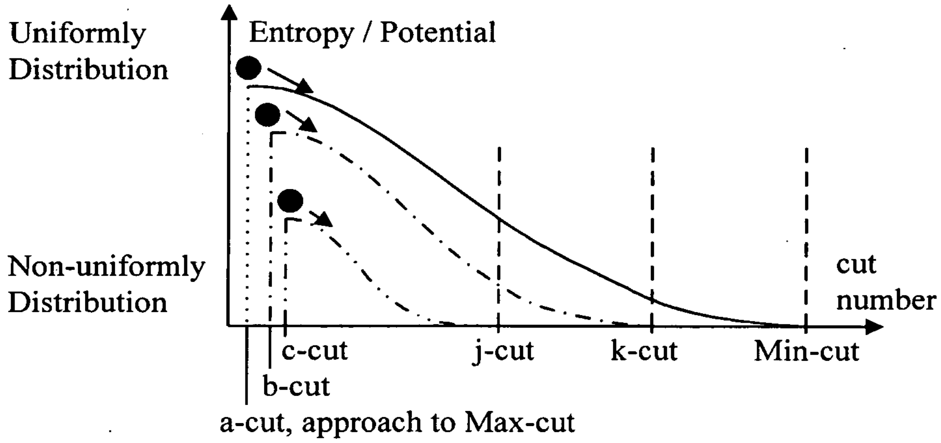


*Fig. 20B*

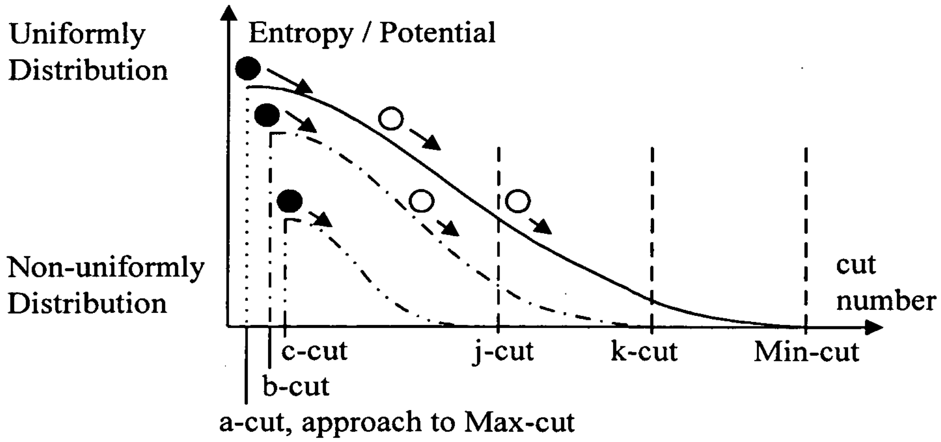


*Fig. 20C*